REMARKS

Please reconsider the application in view of the above amendments and the following remarks. Applicant thanks the Examiner for carefully considering this application.

Information Disclosure Statement

Applicant respectfully requests that the PTO Form-1449 from the IDS filed on December 9, 2005, be initialed and returned. If these Information Disclosure Statements have not been considered, appropriate consideration thereof is respectfully requested.

Disposition of Claims

Claims 1, 3-6, 10, 12-15, and 17 are pending in the present application. Claims 1, 10, and 17 are independent. The remaining claims depend, directly or indirectly, from claims 1, 10, and 17.

Rejection(s) under 35 U.S.C. § 103

Claims 1, 3-4, 6, 10, 12-13, and 15

Claims 1, 3-4, 6, 10, 12-13, and 15 are rejected under 35 U.S.C. § 103(a) as being obvious over Intel® IA-64 "Architecture Software Developer's Manual (hereinafter "IA-64") in view of "A Framework for Balancing Control Flow and Predication" by Hwu *et al.* (hereinafter "Hwu"), further in view of U.S. Patent No. 6,637,026 issued to Chen (hereinafter "Chen"). For the reasons set forth below, this rejection is respectfully traversed.

The present invention is directed to a method for optimizing predicated code, in which an abstract internal representation of code is converted and mapped to a machine representation (*see, e.g.,* Specification, Abstract). The use of new instructions, in combination with the use of a general register designated as a designation register, eliminates the use of condition codes (*see, e.g.*, Specification, page 12, line 19 – page 13, line 5). Further, in one embodiment of the invention, the machine representation is then optimized. Optimization of the machine representation includes, for example, obtaining a signature of a cover for identification (*see, e.g.*, Specification, page 22, line 18 – page 25, line 23).

Accordingly, claims 1 and 10 require if-converting an abstract internal representation, where the if-converting comprises testing a condition code associated with a conditional instruction and writing Boolean data to a *general* register designated as a destination register based on the testing, the destination register corresponding to the conditional instruction and representing a predicate.

Claims 1 and 10 additionally require transforming the if-converted representation to a machine representation. Further, claims 1 and 10 require optimizing the *machine representation* based on a combination of a predetermined cover analysis and a predetermined replacement pattern such that a redundant instruction in the machine representation is eliminated.

IA-64, in contrast to the present invention, does not teach or suggest if-converting that comprises testing a condition code associated with a conditional instruction and writing Boolean data to a *general* register designated as a destination register based on the testing. The Examiner asserts that using a standard register to store predicate Boolean data equates to designating a generic register to make it a predicate register (*see* Office Action dated October 17, 2005, at page 10). However, IA-64 explicitly states that predicates are stored in specialized *predicate registers* (*see* IA-64, Section 4.3.2, first paragraph). Thus, IA-64 does not contemplate the use of a *general* register, and the two systems are fundamentally different.

In contrast to using a predicate register as taught by IA-64, using a general register allows for the elimination of condition codes that degrade instruction-level parallelism. Further, as a result of not contemplating the use of general registers, IA-64 does not contemplate an entire logical instruction set for a register. Rather, IA-64, like other prior art systems, would be limited to processing dedicated predicate registers without full logical instruction support (*see, e.g.*, Specification, page 27, lines 1-6).

Further, IA-64 fails to teach or suggest optimizing the *machine representation* based on a combination of a predetermined cover analysis and a predetermined replacement pattern such that a redundant instruction in the machine representation is eliminated. IA-64, in sections 10.2.1.2, 10.2.3.1, and 10.2.4, purportedly discloses this limitation. However, this is incorrect. IA-64 clearly discloses predication, or if-conversion, from software code (*e.g.*, C code) to assembler to remove branches from program sequences (*see* IA-64, section 10.2.3.1). One skilled in the art would appreciate the clear distinction between optimizing a machine representation and if-converting a software code sequence to a machine representation. Accordingly, IA-64 is wholly silent as to optimizing the *machine representation* as required by the claims.

Hwu, as discussed above with reference to IA-64, does not show or suggest testing a condition code associated with a conditional instruction and writing Boolean data to a *general* register designated as a destination register based on the testing. Further, Hwu does not show or suggest optimizing a *machine representation* based on a combination of a predetermined cover analysis and a predetermined replacement pattern such that a redundant instruction in the machine representation is eliminated. This is evidenced by the fact that Hwu is used only in an attempt to disclose pitfalls of using predication when hardware resources, interference, or repeated use of same resources and scheduling time frame are a prohibitive factor (*see* Office Action dated 10/17/05, at page 3). In clear contrast to the present invention, Hwu is directed to maximizing the benefits of predication while delaying the final balancing of control flow and predication to schedule time (*see* Hwu, abstract).

Chen, as discussed above with reference to IA-64 and Hwu, does not show or suggest testing a condition code associated with a conditional instruction and writing Boolean data to a *general* register designated as a destination register based on the testing. Further, Chen does not show or suggest optimizing a *machine representation* based on a combination of a predetermined cover analysis and a predetermined replacement pattern such that a redundant instruction in the machine representation is eliminated. This is evidenced by the fact that Chen is used only in an attempt to show virtual statement code checking to establish whether a predicate execution is to be taken or eliminated (*see* Office Action dated October 17, 2005, at page 4).

In fact, Chen teaches away from the claimed invention. Like IA-64, Chen explicitly discloses the use of specialized predicate registers to hold predicates **132, 134** (*see* Chen, col. 3, lines 4-10). Like IA-64, Chen does not contemplate the use of a *general* register, or writing Boolean data to a general register designated as a destination register. Further, Chen's teaching of the use of specialized predicate registers is squarely in contrast with, and teaches away from, the claimed invention.

In view of the above, IA-64, Hwu, and Chen, whether taken separately or in combination, fail to show or suggest the present invention as recited in independent claims 1 and 10. Thus, independent claims 1 and 10 are patentable over IA-64, Hwu, and Chen. Claims 3, 4, 6, 1, 13, and 15, directly or indirectly dependent from claims 1 and 10, are allowable for at least the same reasons. Accordingly, withdrawal of this rejection is respectfully requested.

Claims 5, 14, and 17

Claims 5, 14, and 17 are rejected under 35 U.S.C. § 103(a) as being obvious over IA-64, Hwu, Chen, and further in view of U.S. Patent No 5,999,738 issued to Schlansker *et al.* (hereinafter "Schlansker"). For the reasons set forth below, this rejection is respectfully traversed.

As discussed above, the claimed invention is directed to a method for optimizing predicated code in which an abstract internal representation of code is converted and mapped to a machine representation, and the machine representation is optimized. Claims 1, 10, and 17 require if-converting an abstract internal representation, where the if-converting comprises testing a condition code associated with a conditional instruction and writing Boolean data to a *general* register designated as a destination register based on the testing, the destination register corresponding to the conditional instruction and representing a predicate. Claims 1, 10, and 17 additionally require transforming the if-converted representation to a machine representation, and optimizing the *machine representation* based on a combination of a predetermined cover analysis and a predetermined replacement pattern such that a redundant instruction in the machine representation is eliminated.

As discussed above, IA-64, Hwu, and Chen do not show or suggest the above limitations of claims 1 or 10. Claim 17 contains limitations similar to claims 1 and 10. Schlansker, as discussed above with reference to IA-64, Hwu, and Chen, does not show or suggest the above limitations of claims 1, 10, and 17, or that which IA-64, Hwu, and Chen lack. This is evidenced by the fact that Schlansker is used only to show compensation code to stand for predicate code (*see* Office action dated October 17, 2005, at page 8). In contrast to the claimed invention, Schlansker is directed to a technique for flexible scheduling of non-speculative instructions such that any non-speculative instruction may be executed before any other non-speculative instruction in a code sequence (*see* Schlansker, Abstract). Like IA-64, Hwu, and Chen, Schlansker is completely silent with respect to writing Boolean data to a general register, or to optimizing a machine representation such that a redundant instruction in the machine representation is eliminated.

In view of the above, IA-64, Hwu, Chen, and Schlansker, whether taken separately or in combination, fail to show or suggest the present invention as recited in independent claims 1, 10, and 17. Thus, independent claims 1, 10, and 17 are patentable over IA-64, Hwu, and Chen. Claims 5 and 14, directly or indirectly dependent from claims 1 and 10, are allowable for at least the same reasons. Accordingly, withdrawal of this rejection is respectfully requested.

Conclusion

Applicant believes this reply is fully responsive to all outstanding issues and places this application in condition for allowance. If this belief is incorrect, or other issues arise, the Examiner is encouraged to contact the undersigned or his associates at the telephone number listed below. Please apply any charges not covered, or any credits, to Deposit Account 50-0591 (Reference Number 03226/037001; P5009).

Dated: January 13, 2006

Respectfully submitted,

By__________________
Thomas K. Scherer
Registration No.: 45,079
OSHA · LIANG LLP
1221 McKinney St., Suite 2800
Houston, Texas 77010
(713) 228-8600
(713) 228-8778 (Fax)
Attorney for Applicant

131931_1